Practical 3: Variable selection

Garyfallos Konstantinoudis

Spring Term 2025

Part 1: The epigenetic clock: Epigenetic marks associated with ageing

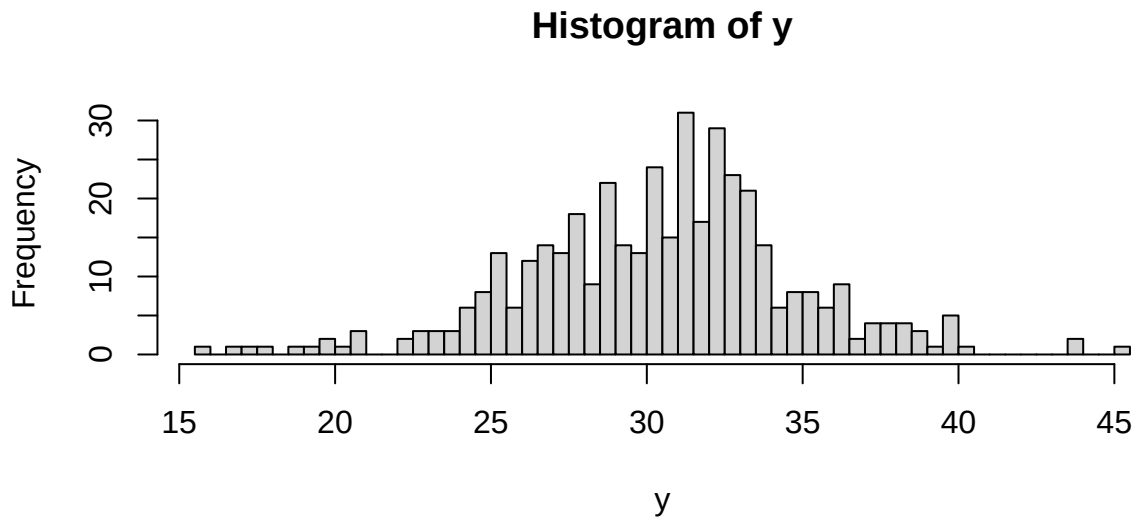
Our epigenome is highly impacted by environmental changes. One particular interesting aspect is ageing and how epigenetic marks such as methylation are affected by ageing. Scientists have shown that there exist specific methylation sites that correlate with age, an observation that has been made in humans, chimpanzees, mice, or rats. Based on our knowledge of which methylation sites correlate with healthy ageing we can use these epigenetic marks as biomarkers to predict the "actual biological age" of an individual. For example, if an individual has been exposed to pollutants or suffered from stress, its "actual biological age" of its body might be much older than its true age.

If you want to read more on the topic, there is a Nature Feature on the scientist Steve Horvath who first proposed to use methylation to measure the biological age

https://www.nature.com/news/biomarkers-and-ageing-the-clock-watcher-1.15014

In this practical we consider data on $n = 409$ healthy mice and methylation of $p = 3,663$ conserved methylation sites. Load the dataset, that contains the methylation matrix as predictor matrix and the age of the mice (in months) stored in the vector y. Familiarise yourself with the dataset using the following commands

```
load("../data/data_epigenetic_clock_control")
#alternatively try load("../data/data_epigenetic_clock_control.rds")
y = control_mice$y_control
hist(y,breaks=50)
```



```r
x = control_mice$x_control
dim(x)
```

```
## [1]  409 3663
```

The first part is concerned with performing a ranking of methylation sites that have the strongest association with ageing.

Question 1.1

Compute a linear regression of the first methylation site against the age of the mice. Note in order to access the first column of a matrix, use square bracket like this [,1] and for the jth variable use [,j]. Figure out which element in the $coefficients matrix contains the p-value of the regression coefficient. Use again the squared brackets to index only the $p-$value.

Question 1.2

In order to compute the massively univariate linear regression estimate, we need to automate this computation for all $p = 3,663$ methylation sites. First initiate a vector where to save your $p-$values

```r
pvec = rep(NaN, 3663)
```

Write a 'for loop' to iterate through all variables. In case you are not familiar with the 'for loop', use this practical here for help https://www.r-bloggers.com/how-to-write-the-first-for-loop-in-r/ . In each iteration j save the $p-$value of the respective regression into the pvec vector at position pvec[j].

Question 1.3

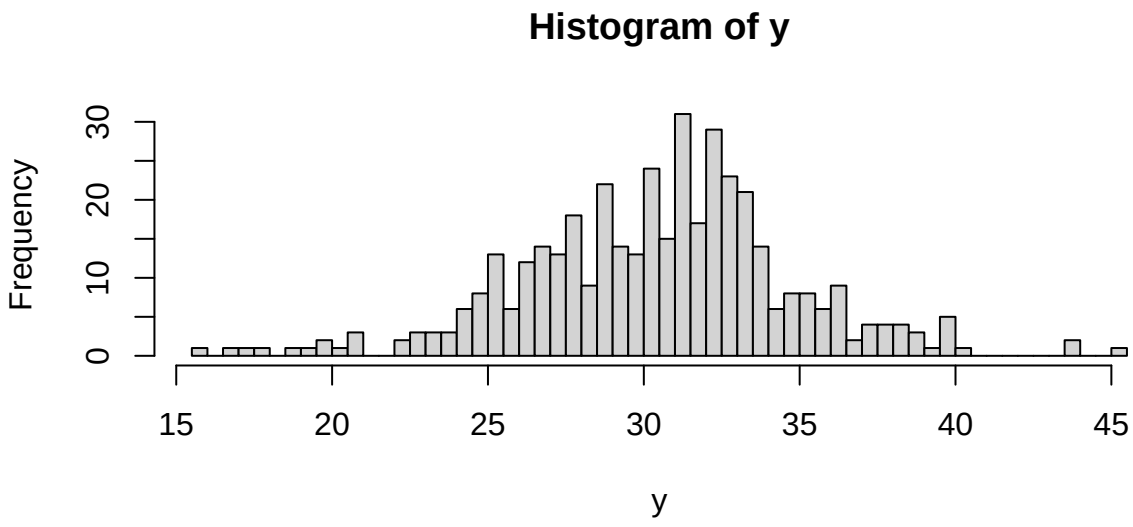
Rank the methylation sites according to their $p-$values and show the top 10 methylation sites that are associated with ageing.

Part 2: The epigenetic clock: A predictive signature for ageing using penalised regression

In the second part of this practical we consider again the same data on $n = 409$ healthy mice and methylation of $p = 3,663$ conserved methylation sites as last week. Our goal is to train our own epigenetic clock and use the methylation data to predict the biological age of mice.

Load the dataset, that contains the methylation matrix as predictor matrix and the age of the mice (in months) stored in the vector y. Familiarise yourself with the dataset using the following commands

```r
load("../data/data_epigenetic_clock_control")
#alternatively try load("../data/data_epigenetic_clock_control.dms")
y = control_mice$y_control
hist(y,breaks=50)
```




```
x = control_mice$x_control
dim(x)
```

```
## [1]  409 3663
```

Question 2.1

First load the glmnet package and fit a lasso regression, where you use y as the outcome and x as predictor matrix (always make sure x is a matrix and not a dataframe). For the first question set lambda to a fixed value equal 0.9. Run the lasso and see how many beta-coefficient are unequal to zero and thus included in the model.

```
library(glmnet)
```

Question 2.2

When performing penalised regression it is not advised to set the regularisation parameter before seeing the data. It is good practice to perform cross-validation (cv) to set the regularisation parameter. Use the cv.glmnet function and the option type.measure = "mse" to optimise the mean squared error (mse). Find the lambda parameter that minimises the cv mse using the value $lambda.min. What is the lambda parameter that is largest, but has a mse that is within one standard error of the minimum mse using the value $lambda.1se?

Use set.seed to fix the random number generator and replicate the same results.

```
library(glmnet)
set.seed(12)
```

Question 2.3

Fit the two lasso models, one with the lambda that optimises the mse, the second with the largest lambda that is within one standard error of the minimum mse. How many variables are included in each model?

Question 2.4

The function cv.glmnet fits regularised regression models for a grid (default length is 100) of different regularisation parameters. The regularisation parameters are stored as $lambda. Additionally cv.glmnet provides the mean cross-validated error ($cvm) and the number of non-zero coefficients ($nzero). Do three plots:

- Plot the sequence 1:100 on the x-axis against the regularisation parameter on the y-axis.
- Plot the sequence 1:100 on the x-axis against the mean cross-validated error on the y-axis.
- Plot the sequence 1:100 on the x-axis against the number of non-zero coefficients on the y-axis.

How do you interpret these plots?

Question 2.5

Fit a cv to define the optimal regularisation for the ridge regression. What are the optimal lambda parameter for the minimum cv mse and 1 standard error within the minimum? Fit a ridge regression with the respective parameter.

Use set.seed for reproducability

```
set.seed(123)
```

Question 2.6

Fit a cv to define the two optimal regularisation parameter for elastic net regression. Focus on the largest lambda within 1 standard error of the minimum. This will provide the sparsest model (fewest predictors) that is almost as good as the one with the minimum cv mse. What are the optimal lambda and alpha parameter? Use the foreach package and the following code to search the optimal combination of lambda and alpha on a grid.

```
set.seed(1234)
library("foreach")
a = seq(0.05, 0.95, 0.05)
search = foreach(i = a, .combine = rbind)%do%{
            cv = cv.glmnet(x,y,family = "gaussian", type.measure = "mse", alpha = i)
            data.frame(cvm = cv$cvm[cv$lambda == cv$lambda.1se], lambda.1se = cv$lambda.1se, alpha = i)
}
elasticnet.cv = search[search$cvm == min(search$cvm), ]
elasticnet.cv
```

```
##        cvm lambda.1se alpha
## 3 10.99065   0.565055  0.15
```

Finally fit an elastic net model using the optimal lambda and alpha regularisation parameter.

Part 3: Which of the 3 models (ridge, lasso and elastic net) builds the better prediction rule?

In the second part we perform a cv to compare how well the three different models can predict new data. To this end we use the

```
library(crossval)
```

package for which we need to write a prediction function. Please see here how to define a prediction function for a linear regression model.

```r
predfun.lm = function(train.x, train.y, test.x, test.y){
    #fit the model and build a prediction rule
    lm.fit = lm(train.y ~ ., data=train.x)
    #predict the new observation based on the test data and the prediction rule
    ynew = predict(lm.fit, test.x )
    #compute mse as squared difference between predicted and observed outcome
    out = mean( (ynew - test.y)^2 )
    return( out )
}
```

Use this code fragment to write a prediction function for the following algorithms:

Question 3.1

Lasso (with the regularisation parameter lambda as set in Question 1.2 to be the largest lambda that has a mse that is within one standard error of the minimum mse using the value $lambda.1se)

Question 3.2

Ridge (with lambda as $lambda.1se in Question 1.5)

Question 3.3

Elastic net (with lambda and alpha as $lambda.1se in Question 1.6)

Question 3.4

Use the crossval package to perform a k-fold cross validation with $k = 5$ folds. For each of the three methods output the mean and standard error of the cv test error and discuss which method generalises best to new data.

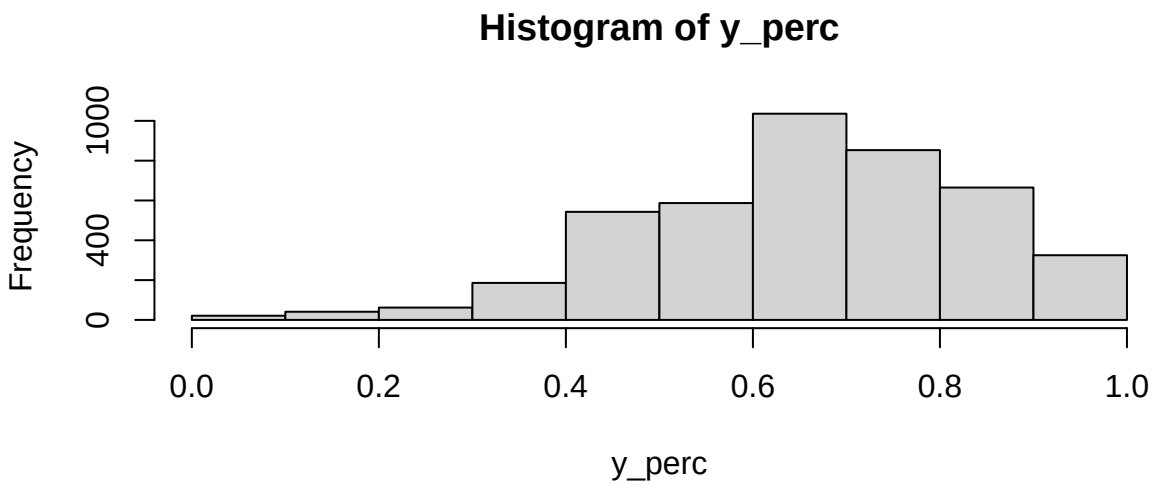
Part 4: Spotify data: Which song features predict if a song is likely to be skipped?

Spotify has created a huge database on characteristics of songs available on spotify. The following data is taken from a Spotify data challenge (https://research.spotify.com/datasets). The data-set contains information on songs, where each song is an observation. Our goal is to define features of a song that predict if a song is likely to get skipped. Load the dataset

```r
load("../data/spotify.Rdata")
```

and familiarise yourself with the outcome variables

```r
y_perc = spotify.data$y_perc
hist(y_perc)
```




```
y_bin = spotify.data$y_bin
table(y_bin)
```

```
## y_bin
## FALSE  TRUE
##   853  3466
```

y_perc is the ratio of the number of skips divided by the number of plays. A value close to one indicates that the song was skipped every time it was played. Please note that y_perc is a quantitative variable, but it is confined in the range of 0 and 1. Consequently, y_perc does not follow a Gaussian distribution. A beta-binomial distribution would be more appropriate. For this practical we are going to focus on y_bin, which is a binary indicator if a song is skipped more than half of the time (y_perc>0.5).

As predictors we consider the variables given in the x_mat matrix. For more information on the features, please see https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/.

```
x_mat = spotify.data$x_mat
colnames(x_mat)
```

```
##  [1] "age"                  "duration"             "us_popularity_estimate"
##  [4] "acousticness"         "beat_strength"        "danceability"
##  [7] "dyn_range_mean"       "energy"               "flatness"
## [10] "instrumentalness"     "liveness"             "loudness"
## [13] "mechanism"            "speechiness"          "tempo"
## [16] "valence"
```

Question 4.1

Build a univariable glm using the glm function where you test each predictor in x_mat at a time in a univariable model with y_bin as outcome. Try to use the for loop to repeat the glm for each predictor. Which feature has the strongest univariable association with y_bin?

Question 4.2

Rank your results by $p-$value add perform a multiple testing correction. To correct for multiple testing, check the function `p.adjust()` and select the `bonferroni` method. How many features are significant after multiple testing?

Question 4.3

Visualise the correlation structure between predictors. Which features are highly correlated?

Question 4.4

Compute a glm using the following command.

```
x_mat = as.data.frame(x_mat)
attach(x_mat)
glm.out = glm(y_bin~age+duration+us_popularity_estimate+acousticness+
                beat_strength+danceability+dyn_range_mean+energy+flatness+
                instrumentalness+liveness+loudness+mechanism+speechiness+
                tempo+valence, family = "binomial")
```

Based on this glm compute the variance inflation factor (car package) and the condition number. Are there signs for multi-collinearity?

Question 4.5

Now perform lasso using glmnet() in the glmnet package to predict the binary outcome y_bin. Perform cross-validation (set.seed(1)) to tune the penalisation parameter and select the largest regularisation parameter within one standard error of the minimum cv-error. Which features are included into the model?

Question 4.6

Finally perform elastic net using glmnet() to predict the binary outcome y_bin. Perform cross-validation (set.seed(2)) to tune the penalisation parameters and select the largest regularisation parameter within one standard error of the minimum cv-error. Which features are included into the model?